TRABAJO FIN DE GRADO

GRADO EN INGENIERIA INFORMATICA

# Título

**Subtítulo**

**Autor**
Francisco de Asís Carrasco Conde

**Director**
María José Rodríguez Fórtiz


Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, Junio de 2025

# Título
# Subtítulo

Francisco de Asís Carrasco Conde

**Palabras clave**: *software libre*

**Resumen**

## Same, but in English

Student's name

**Keywords**: *open source*, *floss*

**Abstract**

D. **María José Rodríguez Fórtiz**, Profesor(a) del Departamento de Lenguajes y Sistemas Informáticos

**Informo:**

Que el presente trabajo, titulado ***Nombre de la App***, ha sido realizado bajo mi supervisión por **Francisco de Asís Carrasco Conde**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a Junio de 2025.

**El/la director(a)/es:**

**María José Rodríguez Fórtiz**

# Agradecimientos

# Índice general

# Índice de figuras

# Índice de tablas

Capítulo 1

# Introducción

## 1.1. Justificación

Primero de todo,¿que me lleva a hacer este TFG? Para empezar, es verdad que hay muchas apps en las stores que son para usuarios de gimnasio, para hacer más fácil el seguimiento y evolución, no obstante, dichas aplicaciones suelen medir parámetros de forma muy general. Me explico, suelen medir calorías quemadas basadas en las pulsaciones (si tienes un reloj inteligente que lo permita) pero este tipo de mediciones de calorías suelen ser poco exactas, me baso en el siguiente estudio Jerath et al., 2023, como añadido a esto no suele ser un dato de mucha relevancia al hacer datos ejericicios de fuerza. Además la gente que suele hacer deporte de manera más informal o por hobbie, no suele tener los conocimientos o aparatos para hacer mediciones de calidad e interpretarlo correctamente.

Otra problemática en el resto de apps, suele ser la falta de accesibilidad, los scrolls abundan, eventos no controlados por el usuario, colores confusos para daltónicos, a veces hay ausencias de iconos y un amplio etcétera. También no suele existir un apartado para ver entrenamientos recomendados por otros, como si fuera un foro, la gente suele buscarlo en redes sociales o videos que se encuentran en la red, muchas veces en estos videos se ponen a dar rodeos para rascar más tiempo, así es como el usuario pierde tiempo para que a lo mejor no sea el entrenamiento que el andaba buscando.

Me veo en la necesidad de hacer esta app para dotar de una herramienta simplificada, fácil de manejar, que permita compartir información entre usuarios de forma eficaz y accesible. Simplificada porque la información que manejo se trata de una forma fácil e intuitiva, con un poco de experiencia en el deporte

se sabe que significa cada parámetro y la que no es tan fácil de interpretar, la interpreta el sistema por el usuario. Fácil de manejar, dado que el usuario solo se tendrá que preocupar de por ejemplo, contar cuantas veces levanto una pesa o cuando empiezo y acabo de correr. Compartir información eficazmente, porque en una sección de la app habrá una parte formato red social, que permitirá tanto buscar usuarios que comparten entrenamientos, como los propios entrenamientos en si, acompañados de su descripción en la que el creador da una breve información acerca del entrenamiento. Accesible, se evitará en todo lo posible usar elementos que perjudiquen a otros usuarios de diversas capacidaddes, entre dichos elementos incluye scrollables y más elementos.

Antes de seguir, me veo en la obligación de hablar un poco sobre conceptos generales en el mundo del deporte. Un ejercicio, es la repetición de un movimiento varias veces para estimular uno o varios músculos. Los entrenamientos, entendámolos como la colección de distintos ejercicios destinados a entrenar una parte o varias del cuerpo, a los entrenamientos también se les llama rutinas. Una serie, es cuando dentro de un ejercicio, repetimos ese movimiento un número determinado de veces y se realiza un descanso, sin cambiar de ejercicio.Una repetición, es la realización del movimiento de ese ejercicio en una serie, es decir, si estoy haciendo flexiones, hago 12, descanso, hago 10, descanso, hago 9 y ya no hago más flexiones, dentro de mi entrenamiento se vería así:

Rutina de entrenamiento 1:

- Ejercicio 1
  - Serie 1: X repeticiones
  - Serie 2: X repeticiones
- Flexiones
  - Serie 1: 12 repeticiones
  - Serie 2: 10 repeticiones
  - Serie 3: 9 repeticiones
- Ejercicio 3
  - Serie 1: X repeticiones
  - Serie 2: X repeticiones

Una vez explicado esto, ¿como voy a dar soporte a los usuarios? De la siguiente forma, midiendo los parámetros que me pida el usuario que pueden ser repeticiones/peso/distancia/tiempo se pueden medir de forma individual o al mismo tiempo. También guardando la cantidad de series de un ejercicio realizadas en un entrenamiento con sus respectivos parámetros. Contabilizandole al usuario el tiempo descansado, para facilitar el correcto entrenamiento del usuario. También si el usuario lo solicita se le enseñará un resumen de su rendimiento

en cada ejercicio con respecto a la fecha en la que se realizó el mismo, para que vea su evolución respecto al tiempo.

También la app ayudará a hacer un control de las metas que se proponga el usuario, es decir, si el usuario se propone bajar de peso o aumentar su rendimiento en x ejercicio, la app le recordará las metas que se autopropuso y le avisará cuando las cumpla.

## 1.2. Objetivos

Los objetivos de este trabajo de fin de grado son:

- Analizar algunas apps del mercado, así como sus capacidades, que ofrecen, su costo tanto para el usuario como para sus creadores y las valoraciones de los usuarios finales, para aclarar que es lo que buscan los usuarios en este tipo de apps y adaptar esas necesidades al software a desarrollar.
- Investigar sobre que herramientas usar para la implementación de la base de datos, backend, frontend y técnicas y metodologías para dar un desarrollo de calidad al software. Parte importante para ayudar a la sostenibilidad del sistema en el tiempo.
- Aprender a usar herramientas actuales y punteras para el desarrollo de aplicaciones.
- Desarrollar una aplicación multiplataforma que cubra todos los requisitos necesarios para dar soporte a la realización de entrenamientos y monitorización de rutinas de ejercicio físico.

## 1.3. Estructura de la Memoria

El 1º capítulo(**Introducción**), es una descripción sobre lo que se va a abordar y que ideas iniciales se tienen acerca del software que se va a desarrollar y algunos aspectos importantes del gremio en el que se usaría.

El 2º capítulo(**Estado del arte**), es un análisis de como está el ámbito sobre el que se va a desarrollar este trabajo, es decir, apps del mercado(sus prestaciones y funciones más importantes) y frameworks que se podrían emplear para el desarrollo, así como sus ventajas y desventajas.

El 3º capítulo,

Capítulo 2

# Estado del arte

### 2.0.1. Aplicaciones similares

Algunas de las aplicaciones que existen, para mediciones de constantes relacionadas durante el entrenamiento físico, suelen ser de pago. Ejemplos de precios mensuales:

- MyFitnessPal: €9.99
- Whoop: €28
- Fitbit Premium: €9.99
- Apple Fitness+: €9.99
- Strava Premium: €5.99

Es verdad que algunas tienen versión gratuita, pero no ofrecen la totalidad de sus funcionalidades, de hecho, estas versiones suelen ser extremadamente limitadas. Otras aplicaciones como Garmin Connect, si que son gratuitas, pero te obligan a ceñirte a un smartwatch de la marca Garmin, los cuales su precio no baja de los 300€. La app presentada daría la mayoría de funcionalidades de pago de una manera gratuita y aporta su funcionalidad de medir de forma personalizada el rendimiento del deportista. También se añade una IA, una funcionalidad que no vista en muchas aplicaciones del sector y si existen, son de pago. La función de la inteligencia artificial sería la de aconsejar al usuario cuando este lo necesite, sobre su rendimiento y/o parámetros medidos durante su entrenamiento.

### 2.0.2. Metodologías potenciales a aplicar

Existen muchas formas de desarrollar software con sus respectivas ventajas, desventajas y forma de trabajar. En este caso, no voy a investigar sobre metodologías que usen la idea de prototipos(software que puede ser probado por el cliente durante el desarrollo):

- **Cascada:** Es una forma de desarrollar de las más clásicas y lineales. Se hace una secuencia de fases de desarrollo, las cuales suelen ser diseño requisitos, diseño, implementación, pruebas, implementación y mantenimiento. Es una metodología fácil de entender y aplicar, eficaz para plazos de entrega estrictos. Sin embargo, no es flexible, si se encuentran fallos en la planificación o alguna de las fases, no hay mucho tiempo de reacción.
- **Desarrollo Ágil:** En lugar de una secuencia rígida de fases, se promuebe la flexibilidad, ya que se combina con reuniones con el cliente para obtener feedback del software funcional desarrollado entregado en dichas reuniones(no es un prototipo ya que solo es una demostración, no se le entrega para su uso durante el desarrollo), permitiendo corregir errores de planificación o en el propio software durante el desarrollo, sin perder mucho tiempo. Es necesario un buen feedback del cliente.
- **Lean Software Development:** Se basa en la idea de descartar todo aquello que no aporte valor para el cliente y quitarle importancia al desarrollo para ganar calidad y sostenibilidad en el tiempo al software. Esto se refleja en que primero se hace una investigación a fondo de todas las herramientas posibles a usar, su consecuente aprendizaje y en lo más tardio de todo esta la elección de que herramientos usar. Una vez hecha la elección, ya se puede empezar con el desarrollo.
- **V Model:** Es como el método cascada, pero antes de pasar a la siguiente fase se realizan las pruebas de la actual y no se procede hasta que estas estén válidas. Sigue teniendo las mismas desventajas que el modelo de cascada, pero es ideal para proyectos en los que es necesario una alta calidad.
- **Modelo en espiral:** En cada fase se realiza planificación, diseño, desarrollo y evaluación de riesgos. Es muy flexible y esta analizando constantemente los riesgos, pero se requiere cierta experiencia para su correcto empleo.

### 2.0.3. Tecnologías potenciales para usar

Para hablar de las tecnologías a emplear, lo voy a separar en las 3 partes importantes de la app la base de datos, backend y frontend.

#### 2.0.3.1. Base de datos

En este punto vamos a hablar de las ventajas y desventajas entre usar BDs relacionales y no relacionales, así como las herramientas a emplear en cada una para desarrollar estos servicios.

**Relacionales:** Las bases de datos relacionales o SQL, como bien sabemos nos permiten guardar datos de una manera bien definida y ordenada, permitiendo asegurar la integridad de la información almacenada. Es muy buena opción, si se prefiere una escalabilidad vertical, es decir, aumentar la capacidad de procesamiento del servidor que va a albergar la BD. Algunas de los SGBDS existentes en la actualidad son:

- MySQL
- Oracle Database
- SQL server(Microsoft)
- PostgreSQL

**No relacionales:** También llamadas NoSQL, sirven para trabajar con estructuras de datos semidefinidas o no definidas. Esto quiere decir que no siguen un esquema rígido, lo cuál puede complicar consultas complejas, pero permiten una gran escalabilidad horizontal, permiten añadir más servidores para que operen con la misma base de datos, por tanto aumentar el número de peticiones que puede atender. Cabe decir que dentro de este tipo de bases de datos existen varios subtipos, cada uno especializado en una cualidad:

- Documentales(MongoDB): guardan los datos en un JSON
- Columnares(Cassandra): los datos no se guardan en filas, sino en columnas ,ideal para tratar grandes volumenes de datos por columnas
- Clave-Valor(DynamoDB): acceso rápido a los datos
- Graficas(Amazon Neptune): para tratar relaciones complejas

#### 2.0.3.2. Backend

En el desarrollo del backend exiten varios frameworks que nos van a permitir un desarrollo rápido, eficaz y de calidad. Nuetro backend se va encargar principalmente de recibir y realizar peticiones de los clientes y/o consultas a la BD. Para ello se han investigado las siguientes herramientas:

- Express.js(Node.js): muy escalable y alto rendimiento

- Django(Python): ideal para sistemas a gran escala
- Ruby on rails: el mejor en velociad de desarrollo

Existen más frameworks, pero he decidido investigar solo sobre aquellos que me permitan un desarrollo más ágil, el resto de herramientas como Spring Boot(Java) o Laravel(PHP), también son buenas herramientas, pero tengo más familiaridad con algunas de las herramientas anteriores.

#### 2.0.3.3. Frontend

Como se dijo en la introducción, uno de los objetivos es que el software sea multiplataforma, así que se investigará sobre frameworks que me permitan esa capacidad:

- Flutter(Dart): de las manos de Google, esta herramienta permite un desarrollo rápido ya que nos permite el hot reload y no tener que recompilar la app cada vez que haya un cambio. También es muy personalizable en lo que respecta a la UI.
- React Native(js): es un framework que extiende React, permitiendo hacer sentir aplicaciones en js como si fueran nativas.
- Xamarín(C#): de Microsoft, esta herramienta es ideal para desarrolladores familiarizados con C# y .NET, también es idóneo para el alto rendimiento y acceso a HW nativo.

### 2.0.4. Trabajos relacionados

El software libre y sus licencias Foundation, s.f. ha permitido llevar a cabo una expansión del aprendizaje de la informática sin precedentes.

Capítulo 3

# Propuesta

## 3.1. Metodología utilizada

Para el desarrollo de esta app, usaré una metodología *ágil* tipo **SCRUM**. He decidido usar esta porque me permite corregir fallos en la velocidad de diseño y/o planificación de forma eficiente y sin dañar el producto final.

Las reuniones con la tutora serán las *sprint reviews*.

## 3.2. Temporización

La temporización se realizó el día 25 de marzo de 2025. La entrega del producto (este TFG) está prevista para el 16 de junio de 2025, es decir, 83 días, o lo que es lo mismo, casi 12 semanas. Si un sprint dura 2 semanas, habrá 6 sprints hasta la entrega final.

La iteración 0 se dedicará al diseño de pantallas y al repaso de las funcionalidades de la app. La idea inicial es dividir la app en varios módulos y centrar cada sprint en uno de ellos:

- Diagramas de la app y ejercicios
- Rutinas, usuarios y sesión
- Datos que ingresa el usuario
- Flujo de entrenamiento

- Inteligencia Artificial (IA)
- Smartwatch

## 3.3. Seguimiento del desarrollo

### Iteración 0

En esta primera iteración me centŕe en completar los diseños de la app, priorizando su accesibilidad. También concreté el *product backlog*, compuesto por 25 historias de usuario. Algunas incluyen tareas secundarias:

**SCRUM-1** Registrar peso por día

**SCRUM-2** Establecer peso objetivo

**SCRUM-3** Insertar/Borrar/Modificar ejercicio de la lista de ejercicios

**SCRUM-4** Buscar rutina en la lista del usuario

**SCRUM-5** Sustituir un ejercicio por otro en la rutina

**SCRUM-6** Insertar/Borrar/Modificar rutina

**SCRUM-7** Poner una meta en cada ejercicio

**SCRUM-8** Hacer grafica de los datos de los ejercicios

**SCRUM-9** Revisar datos para ver si el descanso es necesario

**SCRUM-10** Enseñar datos de una rutina a descargar

**SCRUM-11** Compartir mi rutina

**SCRUM-12** Guardar las repeticiones y series de todos los ejercicios de un entrenamiento

**SCRUM-13** Valorar el entrenamiento en base a la marca actual y la meta del usuario

**SCRUM-14** Monitorizar pulso en tiempo real

**SCRUM-15** Medir pulso en reposo y compararlo con datos de ejercicios

**SCRUM-16** Avisar de anomalías en el pulso de forma suave

**SCRUM-17** Obtener calorías quemadas

**SCRUM-18** Comprobar el equilibrio nervioso del usuario

**SCRUM-19** Realizar el flujo del entrenamiento

**SCRUM-20** Conectar con la IA para iniciar diálogo

**SCRUM-21** Crear/Borrar usuario

**SCRUM-22** Resumir datos

**SCRUM-23** Iniciar/Cerrar sesión

**SCRUM-24** Medir SpO2

**SCRUM-25** Interpretar constantes

**SCRUM-26** Buscar ejercicios en la lista de ejercicios

**SCRUM-27** Hacer la IU de la lista de los ejercicios

**SCRUM-28** Hacer la IU de creación de ejercicio

**SCRUM-29** Pop up de confirmación

**SCRUM-30** Hacer IU de datos ejercicio

**SCRUM-31** Crear IU de la lista de las rutinas

**SCRUM-32** Crear IU de pantalla inicial

**SCRUM-33** Crear IU menú principal

**SCRUM-34** Crear IU datos rutinas

**SCRUM-35** IU lista ejercicios de rutina modificable

**SCRUM-36** Crear IU para crear rutina

**SCRUM-36** Detectar metas cumplidas despues del entrenamiento

**SCRUM-37** Crear IU del perfil del usuario

**SCRUM-38** Detectar metas cumplidas en los ejercicios despues del entrenamiento

**SCRUM-39** Crear IU de buscar rutinas para descargar

**SCRUM-40** IU del pop up de resumir datos

**SCRUM-41** IU desplegable rutina

A continuación se muestran los diseños creados en esta iteración:


Figura 3.1: Pantalla inicial

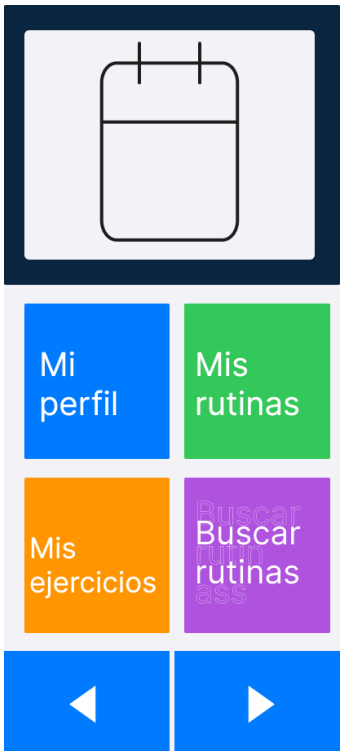


Figura 3.2: Menú principal


Figura 3.3: Lista ejercicios

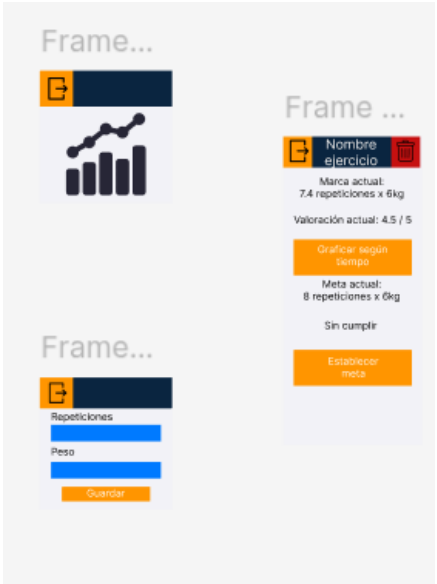


Figura 3.4: Datos ejercicios


Figura 3.5: Lista rutinas

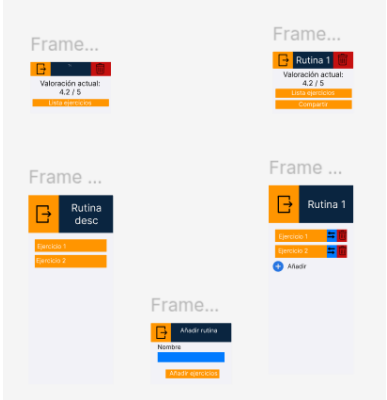


Figura 3.6: Datos rutinas


Figura 3.7: Ajustes de perfil

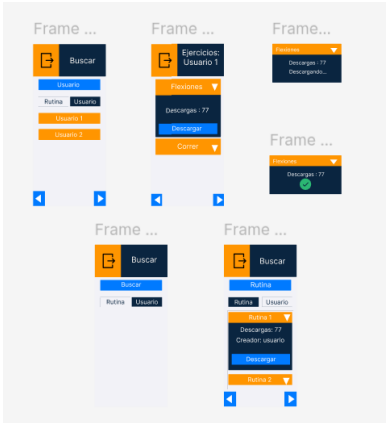


Figura 3.8: Descargar rutinas

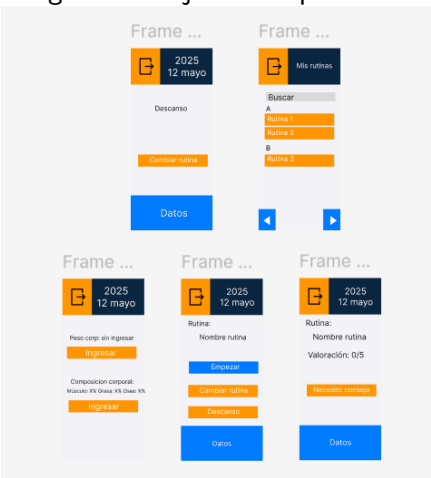


Figura 3.9: Seleccionar entrenamiento


Figura 3.10: Flujo Entrenamiento 0


Figura 3.11: Flujo Entrenamiento 1



Figura 3.12: Flujo Entrenamiento 2 Figura 3.13: Flujo Entrenamiento 3

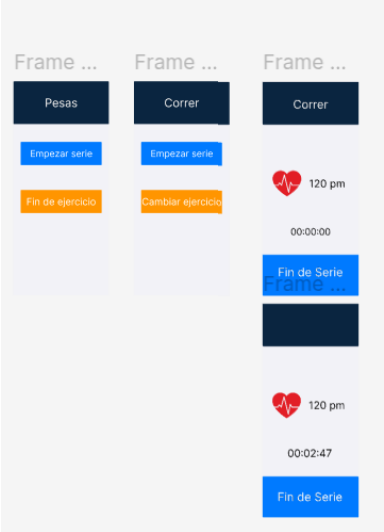


Figura 3.14: Flujo Entrenamiento 4


Figura 3.15: Flujo Entrenamiento 5


Figura 3.16: Flujo Entrenamiento 6

Estos diseños pueden cambiar y sufrir alteraciones debido a los sprints reviews y correcciones de accesibilidad. De todas formas, cada cambio será informado con su motivo.

### Sprint Review 0

Durante esta revisión de sprint se realizaron mejoras de diseño, como la incorporación de iconos en todos los botones para mejorar la accesibilidad. Se revisaron los títulos de las ventanas, cambiando aquellos que no eran lo suficientemente claros.

Además, se añadieron nuevas historias al *product backlog*:

- SCRUM-42: Copiar rutina
- SCRUM-43: Añadir meta por parámetro
- SCRUM-44: Solicitar permiso al usuario antes de enviar datos a la IA
- SCRUM-45: Enviar datos del entrenamiento actual y anteriores a la IA
- SCRUM-46: Conectar/Desconectar con la IA

## Iteración 1

Siendo la lista de la primera iteracion de la siguiente manera:

- SCRUM-26: Buscar ejercicio en lista
- SCRUM-3: Insertar/Borrar/Modificar ejercicio de la lista de ejercicios
- SCRUM-27: Hacer la IU de la lista de ejercicios
- SCRUM-28: Hacer la IU de creación de ejercicio
- SCRUM-29: Pop up de confirmación
- SCRUM-30: Hacer IU de datos ejercicio

Dentro del SCRUM-3 tenemos subfuncionalidades, funciones pequeñas que componen la grande:

- SCRUM-3.1 Borrar de la BD
- SCRUM-3.2 Insertar en la BD
- SCRUM-3.3 Hacer notorios los cambios en vivo

- SCRUM-3.4 Modificar en la BD

Durante esta iteración me familiaricé con Flutter y sus herramientas, como la base de datos local *SQLite*.

#### SQLite

Es una base de datos ligera, autocontenida y de código abierto. Usa el mismo lenguaje de consultas que SQL, lo que la hace ideal para la app. Además, permite transportar toda la BD en un *.db*, lo cual puede facilitar funcionalidades como copias de seguridad descargables desde la nube.

#### Sprint Review 1

Las primeras iteraciones tienden a ser más lentas debido a la fase inicial del desarrollo. No se completó la totalidad del sprint; quedó pendiente la subtarea de modificar ejercicio en la base de datos. Aun así, las expectativas son positivas, ya que se espera un aumento en la velocidad de desarrollo. No obstante, en lo que se lleva desarrollado han aparecido pocas problemáticas

### Iteración 2

Esta iteración se centró en el desarrollo del sistema de sesiones, la API de la aplicación, el backend básico del servidor y la sección de rutinas.

- SCRUM-21 Crear/Borrar mi usuario
- SCRUM-23 Iniciar/Cerrar sesión
- SCRUM-6 Insertar/Borrar/Modificar rutina
- SCRUM-4 Buscar rutina en la lista del usuario
- SCRUM-31 Crear IU de la lista de las rutinas
- SCRUM-32 Crear IU de pantalla inicial
- SCRUM-33 Crear IU menú principal
- SCRUM-34 Crear IU datos rutinas
- SCRUM-35 IU lista ejercicios de rutina modificable
- SCRUM-36 Crear IU para crear rutina


Figura 3.17: Pantalla nueva

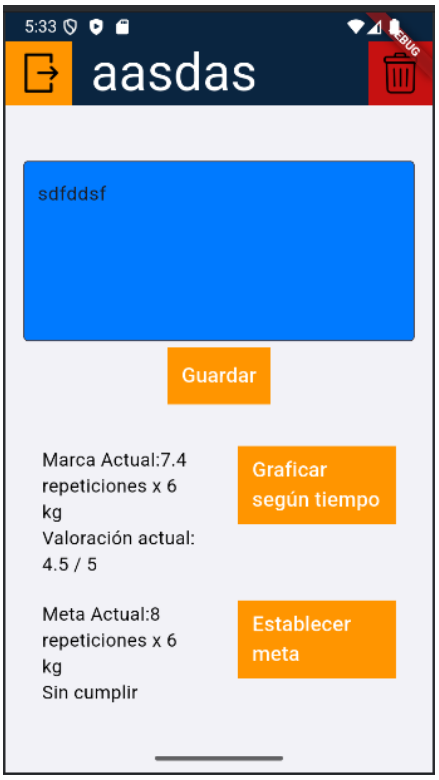


Figura 3.18: Pantalla antigua

Figura 3.19: Comparación entre la pantalla nueva y la anterior

Subtareas de esta iteración:

- SCRUM-25.1 Iniciar sesion
- SCRUM-25.2 Crear cuenta 1
- SCRUM-25.3 Crear cuenta 2

### Sistema de sesiones

Se implementó un sistema de autenticación basado en tokens. Cuando el usuario verifica su identidad, recibe un token que se guarda en el dispositivo y se usa para validar el acceso posterior. Esto evita que usuarios no autorizados suban contenido haciéndose pasar por otros.

El backend está desarrollado en *Node.js* y utiliza una base de datos SQL para gestionar las contraseñas de los usuarios. Este mismo backend se usará para las funcionalidades de las siguientes iteraciones.

#### API de la app

La comunicación entre la app y el servidor se realiza mediante peticiones HTTP. En el futuro, estas podrán migrarse a HTTPS para proteger operaciones sensibles. Por ahora se mantiene HTTP para facilitar el depurado.

#### Cambios importantes

Se rediseñó la implementación de las pantallas con listas. En lugar de una única clase con condiciones para la visibilidad de elementos, se optó por crear clases específicas para cada tipo de lista.

**Ventaja:** El código es más limpio, escalable y modular. Si hay que modificar un tipo de lista, los demás no se ven afectados.

#### Sprint Review 2

Se sugirieron mejoras menores como el ajuste de tamaños e iconos en los botones de la sección de creación de rutinas.

También se modificó la funcionalidad de compartir rutinas. Ahora todas son modificables, eliminando la distinción entre rutinas descargadas (antes no modificables) y creadas (modificables). Esto mejora la experiencia del usuario: si desea volver a una rutina original, simplemente la puede volver a buscar y descargar.

Se cumplieron todas las funcionalidades en este sprint.

### Iteración 3

Esta iteración se centró en desarrollar la funcionalidad para compartir rutinas entre usuarios, incluyendo la subida, visualización y descarga de rutinas. Además, se comenzó a implementar la funcionalidad de resumen de datos para optimizar el uso de la memoria local.

- SCRUM-2 Establecer peso objetivo
- SCRUM-37 Crear IU del perfil del usuario
- SCRUM-42 Copiar rutina

- SCRUM-10 Enseñar datos de una rutina a descargar
- SCRUM-11 Compartir mi rutina
- SCRUM-39 Crear IU de buscar rutinas para descargar
- SCRUM-9 Revisar datos para ver si el descanso es necesario
- SCRUM-22 Resumir datos
- SCRUM-40 IU del pop up de resumir datos
- SCRUM-41 IU desplegable rutina

#### 3.3.0.1. Funcionalidad descartada

Se ha decidido, descartar una funcionalidad, en concreto el SCRUM-9 (Revisar datos para ver si el descanso es necesario), dado porque necesitaría los datos de un smartwatch y ha sido una funcionalidad previamente descartada.

También se ha descartado SCRUM-42 (Copiar rutina), dado que aporta poco valor a la app.

### Peso objetivo

El peso objetivo es una meta que define el usuario y se almacena en memoria. Esta meta se representará en la gráfica de evolución del peso del usuario, y se generará una alerta cuando dicha meta sea alcanzada. La visualización aún no ha sido implementada completamente.

### Cambios durante el desarrollo

Durante esta iteración surgieron cambios en la forma de visualizar y gestionar las rutinas compartidas y almacenadas:

- Ahora, al descargar una rutina, se añade el nombre del creador al nombre de la rutina, facilitando la distinción entre rutinas propias y descargadas.
- Cuando hay conflicto de nombres al crear una rutina, se genera un nombre alternativo del tipo `Nombre(n)`, siendo `n` un número incremental.
- Se ha implementado un selector entre "Mis rutinas" y "Compartidas" para facilitar la visualización de las rutinas subidas a la nube por el usuario.
- Se reemplazó el filtro integrado en la ventana de búsqueda por un selector emergente que permite buscar rutinas por nombre o por nombre de usuario.

- La interfaz de descarga y visualización de rutinas ahora es un cuadro emergente (pop-up) en lugar de un desplegable.

### Imprevistos y problemas en el desarrollo

Se identificaron dos errores principales de planificación:

**Error 1:** Se planificó implementar la funcionalidad de resumir datos sin haber completado la obtención de datos.
**Solución:** Replanificar los sprints.

**Error 2:** Se subestimó la complejidad del sistema de rutinas compartidas, especialmente en la resolución de conflictos de nombres.
**Solución:** Aplicar nombres combinados (nombre + creador + copia) en memoria local e identificadores *autoincrementales* en la nube.

### Valor añadido a la app

Los imprevistos y problemas surgidos durante el desarrollo han permitido detectar *bugs*, errores de diseño y carencias funcionales que de otro modo podrían haber pasado desapercibidos. Gracias a ello, se han podido proponer nuevas funcionalidades y mejorar las ya existentes, lo cual contribuye significativamente a aumentar la calidad global de la aplicación.

Algunas de las funcionalidades propuestas como mejora son:

- Verificar el token del usuario antes de permitir la subida de rutinas, para evitar suplantación.
- Al seleccionar la opción de borrar usuario del dispositivo, eliminar también la base de datos local asociada.
- Crear una base de datos local independiente para cada nuevo usuario creado en un dispositivo.
- Posibilidad de editar el nombre de una rutina.
- Marcar los ejercicios eliminados con una *flag*, para evitar que se inicien rutinas que los contengan.
- Marcar a los usuarios permanentemente eliminados con una *flag*, para proceder a eliminarlos en los dispositvos.

Estas funcionalidades están pensadas para aportar mayor **seguridad**, **integridad** y **calidad** al producto final.

#### 3.3.0.2. Base de datos del backend

La base de datos utilizada en el backend está basada en *MySQL*. En ella se almacenan los usuarios junto con sus contraseñas, así como los ejercicios y las rutinas. Las rutinas están vinculadas al usuario que las creó, y los ejercicios se asocian a las rutinas a las que pertenecen. Si un usuario decide eliminar su cuenta de forma permanente, las rutinas y ejercicios relacionados se eliminarán en cascada.

Al descargar una rutina, los ejercicios asociados se guardan en la memoria local del usuario como si fueran de su propiedad. Es decir, el usuario puede modificarlos libremente sin restricciones.

#### 3.3.0.3. Sprint Review 3

En esta review se estuvo debatiendo si quitar las funcionalidades relacionadas con el smartwatch, por cuestiones de tiempo, ya que con las funcionalidades añadidas en el capitulo anterior se va ajustando el plazo.

En esta iteración se tenía pensado implementar la funcionalidad de resumir los datos, no obstante, al no tener implementada la parte en la que guardo las marcas de los entrenamientos, no puedo trabajar en esa funcionalidad. Por lo tanto, se ha pospuesto para la próxima iteración. También se le dió el visto bueno a la implementación de las funcionalidades previamente expuestas durante el desarrollo.

También, debido al querer hacer la app accesible se ha decidido sustituir los pop ups por pantallas normales, dado que los pop ups son un gran problema para este objetivo, debido a que un usuario puede hacerlo desaparecer este tocando fuera del area en pantalla de este.

### 3.3.1. Iteración 4

En esta iteración se tiene pensado implementar la funcionalidad del calendario, donde el usuario podrá ver su planificación y acceder a sus marcas. Por otro lado, también se implementarán las funcionalidades nuevas del sprint anterior y las nuevas pantallas.

- SCRUM-39 Detectar metas cumplidas en los ejercicios despues del entrenamiento
- SCRUM-1 Registrar peso por día
- SCRUM-19 Realizar el flujo del entrenamiento
- SCRUM-43: Añadir meta por parámetro

- SCRUM-44: Implementar calendario

En esta iteración hay pocas funcionalidades commparadas con otras, porque el flujo del entrenamiento es una funcionalidad muy grande. A parte se añade esta funcionalidad que también se va a realizar en esta iteración, SCRUM-44 Implementar calendario.

#### 3.3.1.1. Paquete tableCalendar

Es un paquete de flutter, que permite implementar y customizar un calendario, de forma fácil y rápida. También permite trabajar con información para reflejarla en las fechass del calendario.

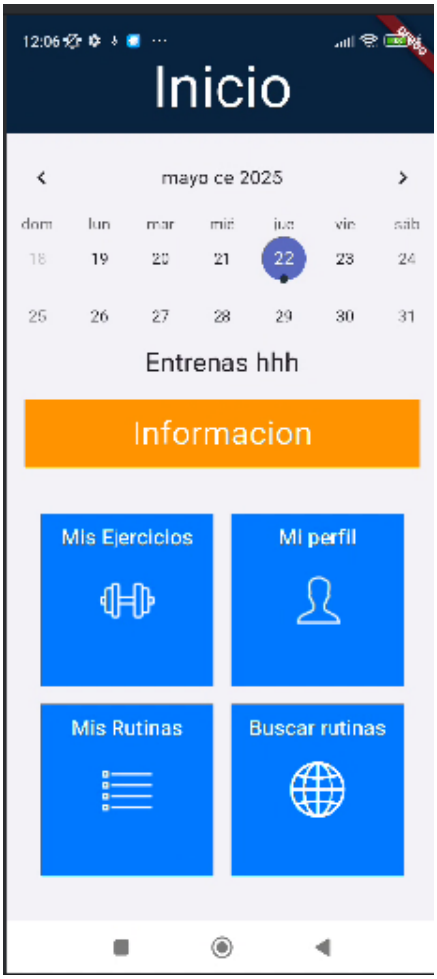


Figura 3.20: Pantalla Calendario

Existen varios formatos de calendario preimplementados, que cambian el numero de dias en pantalla, se escogió este porque es el que menos problemas da en pantalla y más cómodo para las interacciones que busca el usuario.

#### 3.3.1.2. Pantallas cambiadas por pop ups

La solución que se ha dado para que este problema no quite mucho tiempo de desarrollo es la siguiente, reciclar todo el contenido posible del interior de los pop ups y plasmarlo tal cual en una pantalla. Dado que los diseños implementados eran buenos, quitando algún fallo que puedan dar por este cambio de formato.

Los pop ups que se cambiaron son los siguientes:

- Los pop ups de confirmacion
- Crear ejercicios
- Modificar ejercicios
- Nueva meta en un ejercicio
- Crear rutinas
- Modificar rutinas
- Modificar ejercicios de una rutina
- Información de una rutina a descargar
- Modificar informacion del perfil del usuario

Por consecuencia las futuras funcionalidades en la que se mencionan los pop ups, pasaran a ser pantallas completas.

#### 3.3.1.3. Cambios en el backend

Los cambios en el backend están relacionados con las siguientes funcionalidades propuestas en la iteracion anterior:

- Cambio 1: Verificar el token del usuario antes de permitir la subida de rutinas, para evitar suplantación.
- Cambio 2: Marcar a los usuarios permanentemente eliminados con una *flag*, para proceder a eliminarlos en los dispositvos.

Cambio 1: la solución implementada es la siguiente, a todas las funciones que puedan ser potencialmente victimas de una suplantación(via peticion normal de http), se les ha añadido un middleware, una función que se ejecuta antes para verificar el token. Express.js permite implementar esto rapidamente

Cambio 2: de primeras se pensó en usar una flag, pero finalmente para evitar perdida de rendimiento al momento de que el usuario realize un entrenamiento(que es cuando en un principio se tenía pensado borrar el ejercicio), se optó por modificar la lista de ejercicios de la rutina al momento de borrar el ejercicio. El número de consultas iba a ser el mismo, pero se ahorra memoria, ya que no añadimos una columna más a la tabla de ejercicios

# Bibliografía

Foundation, F. S. (s.f.). GNU General Public License.

Jerath, R., Syam, M., & Ahmed, S. (2023). The Future of Stress Management: Integration of Smartwatches and HRV Technology. *Sensors*, *23*(17), 7314. https://doi.org/10.3390/s23177314